FINAL REPORT

PARKINSON'S DISEASE SEVERITY PREDICTION USING MACHINE LEARNING

PRESENTED BY:

AMNA AHMAD	335246
MUHAMMAD MUSTAFA	331945
KHUBAIB AHMAD QURESHI	336691

Table of Contents

Project report: Parkinson's Disease Severity Prediction using Machine Learning

Problem statement

Parkinson's disease is a neurodegenerative disorder that affects millions of people worldwide. Early detection and monitoring of the disease are crucial for timely intervention and improved patient care. Traditional methods of Parkinson's disease assessment require clinical visits, which can be time-consuming and inconvenient for patients. Therefore, there is a need for non-invasive and accessible techniques that can accurately detect and monitor the disease.

Abstract

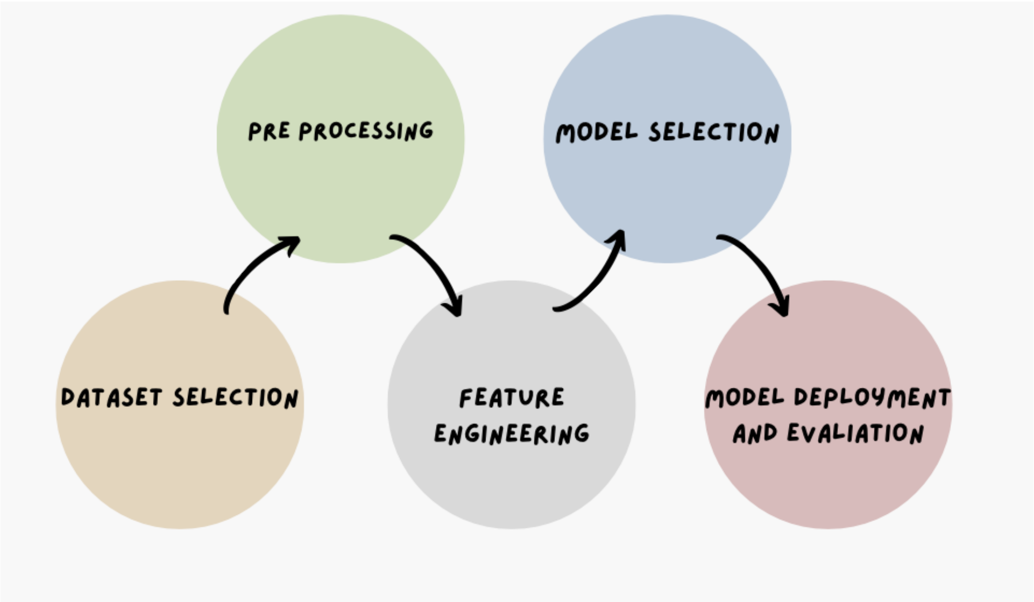
The Parkinson’s Telemonitoring dataset provides a valuable resource for the development of machine learning algorithms for the early detection and monitoring of Parkinson's disease. The dataset consists of voice recordings and clinical information from individuals with Parkinson's disease and healthy controls. The voice recordings were collected using a telemonitoring platform, allowing patients to record their voice while performing simple tasks.

Description

The objective of this project is to develop a machine learning model that can accurately predict the severity of Parkinson's disease in patients based on various voice and speech attributes. The model will be trained on the Parkinson's Disease Dataset, which contains a range of acoustic features that are indicative of the disease severity. The model's performance will be evaluated using various metrics such as accuracy, mean squared error, and R2 score. The ultimate goal of the project is to develop a reliable and accurate tool that can assist doctors in the early detection and diagnosis of Parkinson's disease, potentially leading to improved patient outcomes.

Overall, the Parkinson’s Telemonitoring dataset offers a unique opportunity to advance the field of Parkinson's disease research and contribute to the development of innovative solutions for early detection and monitoring of this debilitating condition.

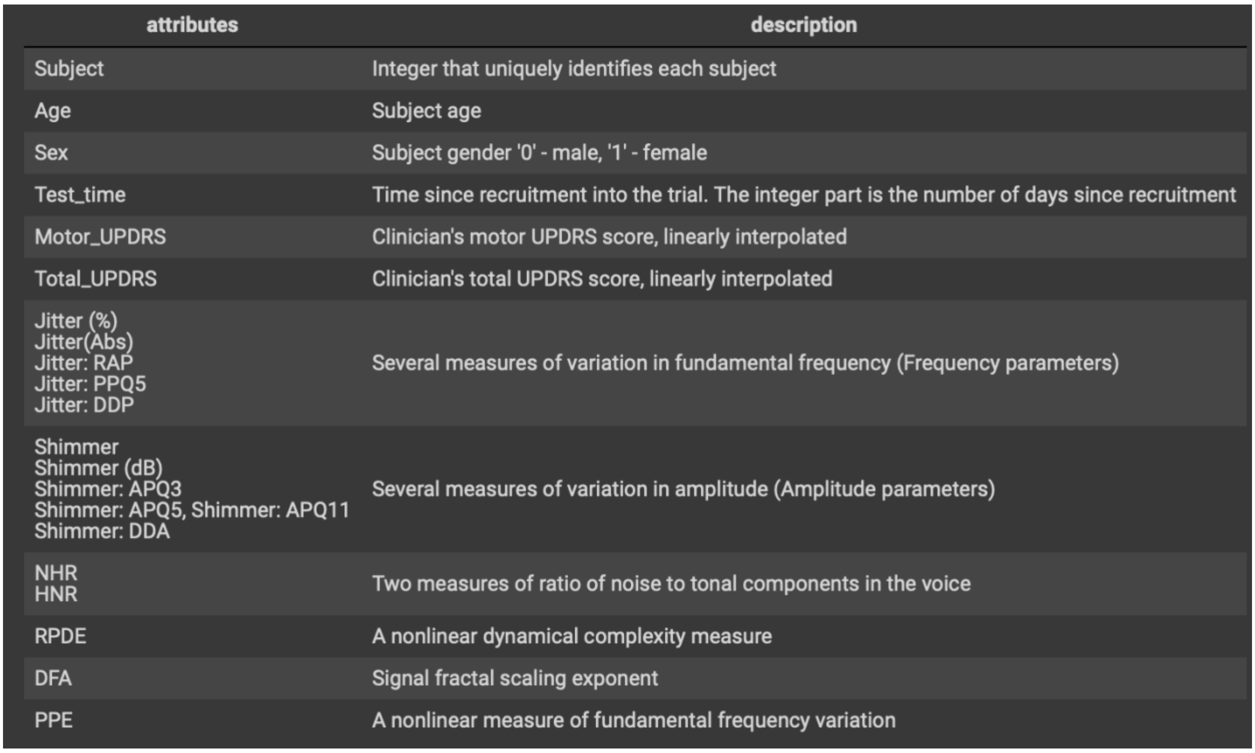
Workflow



Dataset

The Parkinson’s Telemonitoring dataset is a collection of voice recordings and corresponding clinical information from individuals with Parkinson's disease. The dataset was collected using a telemonitoring platform that allowed patients to record their voice while performing simple tasks, such as sustained phonation and vowel phonation. The dataset contains 5875 voice recordings from 42 individuals with Parkinson's disease and 24 healthy controls.

Dataset attributes and their description

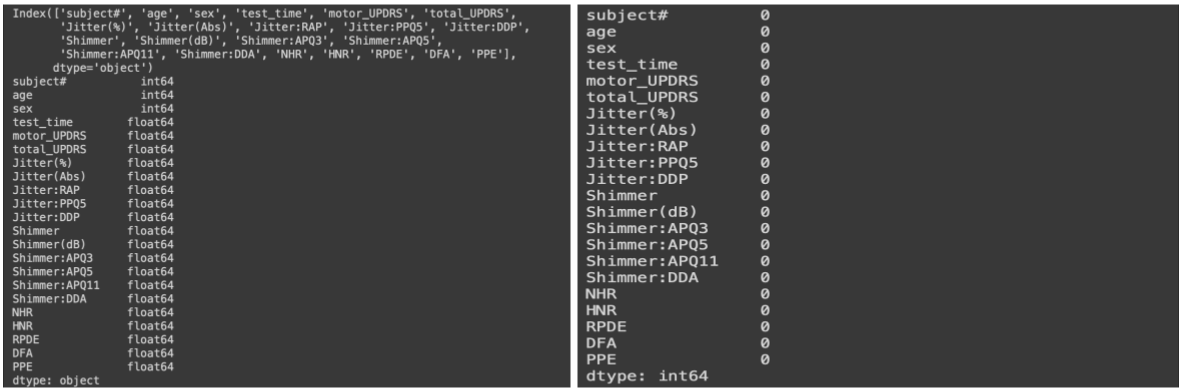


attributes	description
Subject	Integer that uniquely identifies each subject
Age	Subject age
Sex	Subject gender '0' - male, '1' - female
Test_time	Time since recruitment into the trial. The integer part is the number of days since recruitment
Motor_UPDRS	Clinician's motor UPDRS score, linearly interpolated
Total_UPDRS	Clinician's total UPDRS score, linearly interpolated
Jitter (%) Jitter(Abs) Jitter: RAP Jitter: PPQ5 Jitter: DDP	Several measures of variation in fundamental frequency (Frequency parameters)
Shimmer Shimmer (dB) Shimmer: APQ3 Shimmer: APQ5, Shimmer: APQ11 Shimmer: DDA	Several measures of variation in amplitude (Amplitude parameters)
NHR HNR	Two measures of ratio of noise to tonal components in the voice
RPDE	A nonlinear dynamical complexity measure
DFA	Signal fractal scaling exponent
PPE	A nonlinear measure of fundamental frequency variation

Table 1: attributes in dataset

Data types of the attributes and null values

The dataset was pretty clean and had no null values. Hence it had no rows to remove on the basis of null values.



```
Index(['subject#', 'age', 'sex', 'test_time', 'motor_UPDRS', 'total_UPDRS',
       'Jitter(%)', 'Jitter(Abs)', 'Jitter:RAP', 'Jitter:PPQ5', 'Jitter:DDP',
       'Shimmer', 'Shimmer(dB)', 'Shimmer:APQ3', 'Shimmer:APQ5',
       'Shimmer:APQ11', 'Shimmer:DDA', 'NHR', 'HNR', 'RPDE', 'DFA', 'PPE'],
      dtype='object')
subject#           int64
age                int64
sex                int64
test_time        float64
motor_UPDRS      float64
total_UPDRS      float64
Jitter(%)        float64
Jitter(Abs)      float64
Jitter:RAP       float64
Jitter:PPQ5      float64
Jitter:DDP       float64
Shimmer          float64
Shimmer(dB)      float64
Shimmer:APQ3     float64
Shimmer:APQ5     float64
Shimmer:APQ11    float64
Shimmer:DDA      float64
NHR              float64
HNR              float64
RPDE             float64
DFA              float64
PPE              float64
dtype: object
```

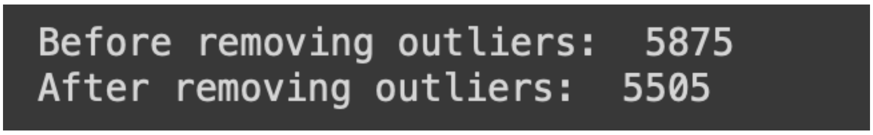
```
subject#         0
age              0
sex              0
test_time        0
motor_UPDRS      0
total_UPDRS      0
Jitter(%)        0
Jitter(Abs)      0
Jitter:RAP       0
Jitter:PPQ5      0
Jitter:DDP       0
Shimmer          0
Shimmer(dB)      0
Shimmer:APQ3     0
Shimmer:APQ5     0
Shimmer:APQ11    0
Shimmer:DDA      0
NHR              0
HNR              0
RPDE             0
DFA              0
PPE              0
dtype: int64
```

Figure 1: data types of attributes and null values

Outliers in the dataset

Outliers in a dataset refer to observations or data points that significantly deviate from the majority of the data. They are values that lie far away from the central tendency of the distribution and can potentially distort statistical analyses and machine learning models.

In the context of the Parkinsons Telemonitoring dataset, outliers can manifest in the clinical measures or acoustic features recorded for each participant. For example, an outlier in a clinical measure might be an unusually high or low value for a particular symptom score, while an outlier in an acoustic feature might be an extreme value for measures like fundamental frequency, jitter, or shimmer.



```
Before removing outliers:  5875
After removing outliers:  5505
```

We used z-scored to remove the outliers from the dataset. Box plots of the features are:

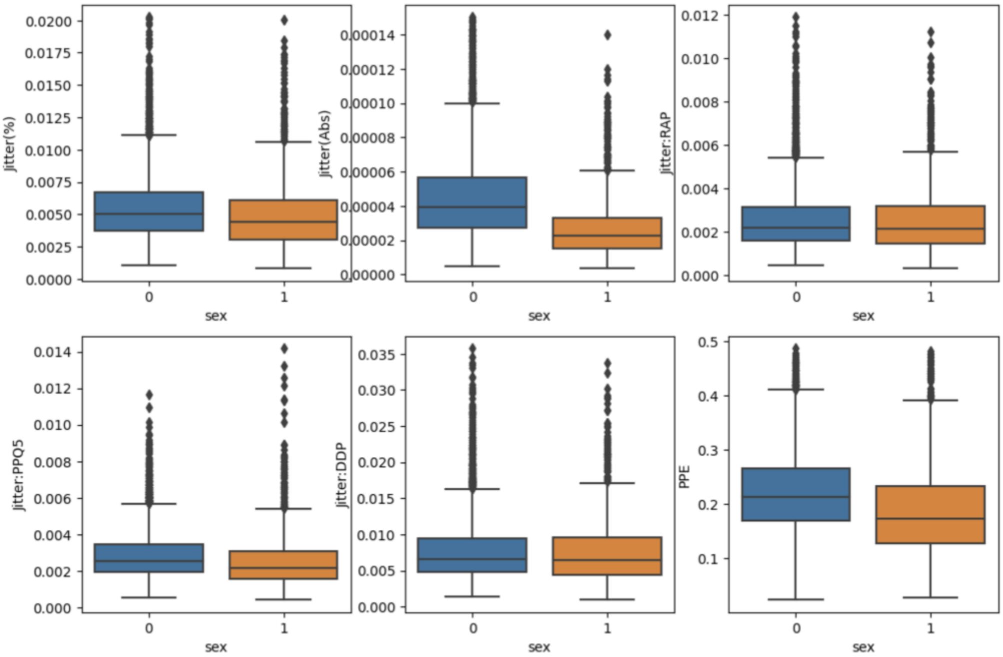


Figure 2: box plots

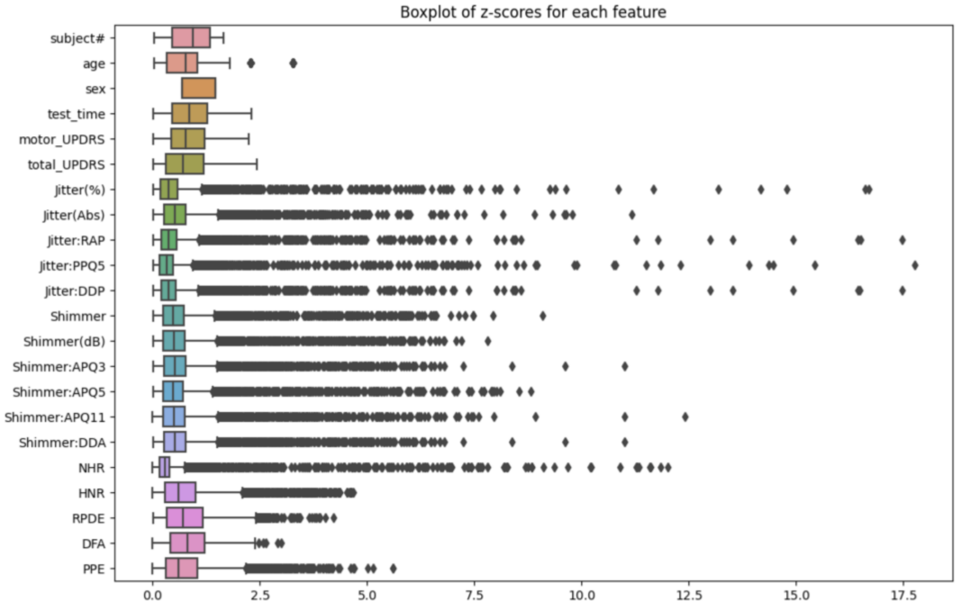


Figure 3: boxplot for every feature for outliers' visualization

Correlation matrix

In the Parkinson's Telemonitoring dataset, the correlation matrix will show the correlation coefficients between all the numerical variables in the dataset, such as age, sex, motor_UPDRS, and total_UPDRS. The coefficients will range from -1 to 1, with 1 indicating a perfect positive correlation, 0 indicating no correlation, and -1 indicating a perfect negative correlation.

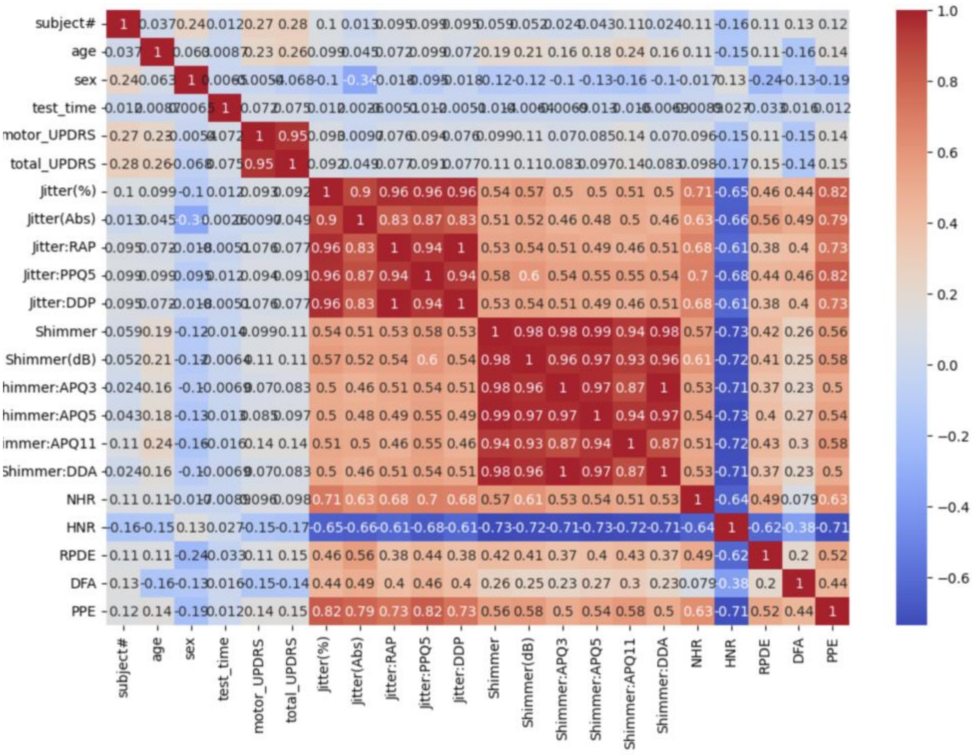


Figure 4: correlation matrix

Data visualization to observe the trends

The following figure shows the age vs count histogram that is used to observe the trends of age an Parkinson. The age 55-65 has highest count thus showing the trends.

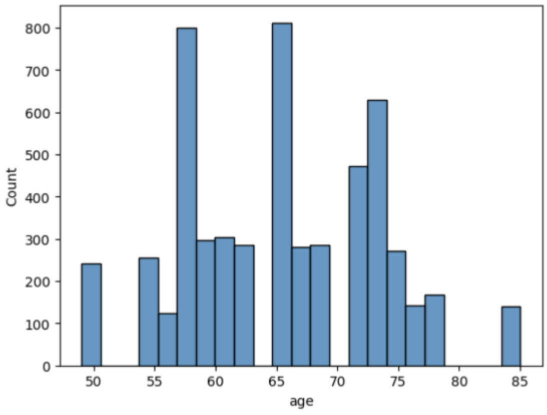


Figure 5:age vs count histogram

The histogram provides insights into how the 'total_UPDRS' scores are distributed across the dataset. It visualizes the frequency of occurrence for different values or value ranges.

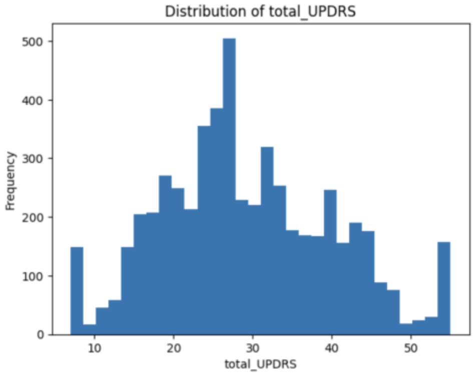


Figure 6 distribution of total_updrs

Scatterplot can reveal whether the relationship between "motor_UPDRS" and "total_UPDRS" follows a linear pattern. If the points in the scatterplot roughly follow a straight line, it suggests a linear relationship.

On the other hand, if the points deviate from a straight line or exhibit a curved pattern, it indicates a non-linear relationship.

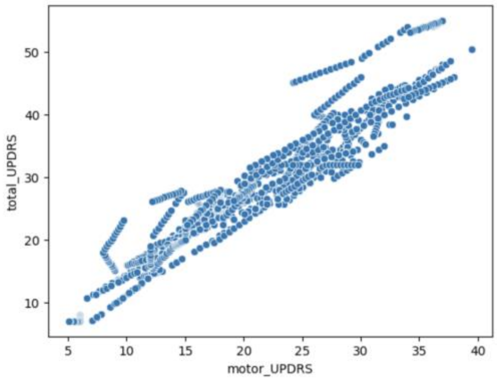
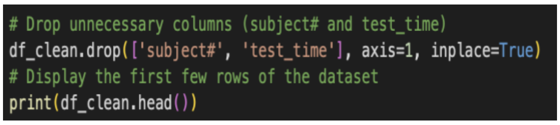


Figure 7 motor vs total _updrs

Feature engineering

Dropping unnecessary columns

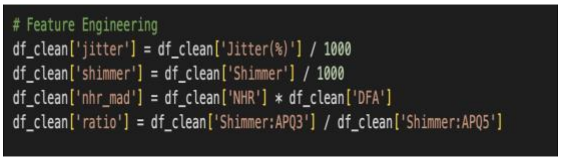
The subject # and test_time attributes were not playing a important role so we , dropped them from the dataset.



```
# Drop unnecessary columns (subject# and test_time)
df_clean.drop(['subject#', 'test_time'], axis=1, inplace=True)
# Display the first few rows of the dataset
print(df_clean.head())
```

Normalization of the values

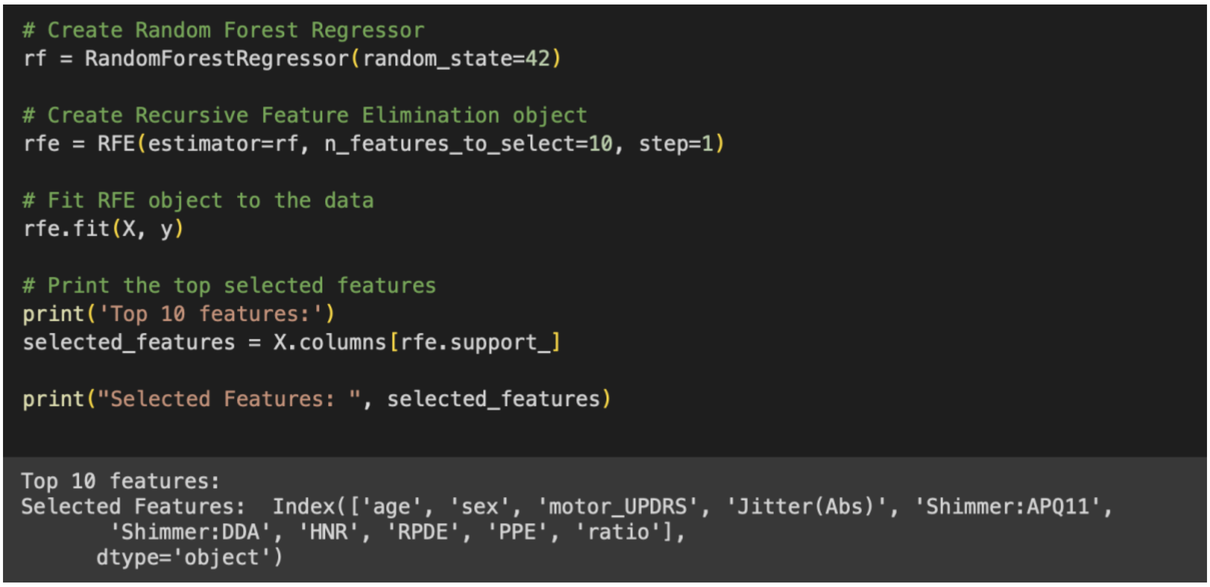
Jitter, shimmer and nhr_mad attributes were normalized and a new attribute ratio was introduced which is basically shimmerAPQ3/shimmerAPQ5



```
# Feature Engineering
df_clean['jitter'] = df_clean['Jitter(%)'] / 1000
df_clean['shimmer'] = df_clean['Shimmer'] / 1000
df_clean['nhr_mad'] = df_clean['NHR'] * df_clean['DFA']
df_clean['ratio'] = df_clean['Shimmer:APQ3'] / df_clean['Shimmer:APQ5']
```

Random forest regressor for feature elimination

Apart form the target variable Total_UPDRS, all other were put into recursive function to list the top 10 features of the dataset.



```
# Create Random Forest Regressor
rf = RandomForestRegressor(random_state=42)

# Create Recursive Feature Elimination object
rfe = RFE(estimator=rf, n_features_to_select=10, step=1)

# Fit RFE object to the data
rfe.fit(X, y)

# Print the top selected features
print('Top 10 features:')
selected_features = X.columns[rfe.support_]

print("Selected Features: ", selected_features)
```

```
Top 10 features:
Selected Features:  Index(['age', 'sex', 'motor_UPDRS', 'Jitter(Abs)', 'Shimmer:APQ11',
       'Shimmer:DDA', 'HNR', 'RPDE', 'PPE', 'ratio'],
      dtype='object')
```

Figure 8 : top 10 features of the dataset

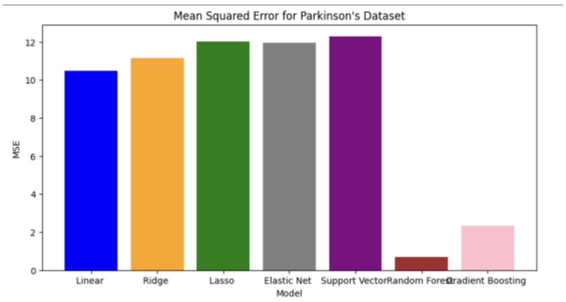
Model selection

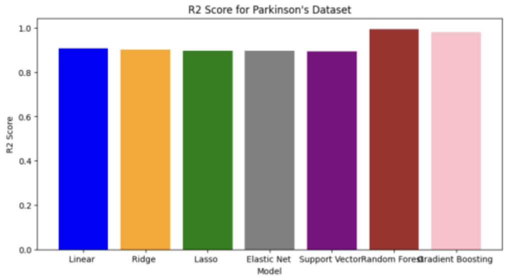
We performed various models on the dataset with and without feature-engineering and results were plotted. The decision was made on the basis of R2-score and mean score.

It splits the data into training and test sets, and trains and evaluates various regression models on the training set. The models include linear regression, ridge regression, lasso regression, elastic net regression, support vector regression, random forest regression, and gradient boosting regression. For each model, the code prints out the mean squared error (MSE) and R-squared (R2) score on the test set. The model with **least mean square error** and **highest R-squared score** will be the best performing model.

Without feature engineering

Model	R2-score	Mean squared error
Linear	R2 = 0.9086697229992886	MSE = 10.489209117005807
Ridge	R2 = 0.9030433271027838	MSE = 11.13539617645207
Lasso	R2 = 0.8953055677799109	MSE = 12.024071633267374
Elastic Net	R2 = 0.8959736973836196	MSE = 11.947337483752982
Support Vector	R2 = 0.8930471900643903	MSE = 12.283444503921178
Random Forest	R2 = 0.9740897326370245	MSE = 0.6787894698620189
Gradient Boosting	R2 = 0.9795612944594523	MSE = 2.3473689507592392



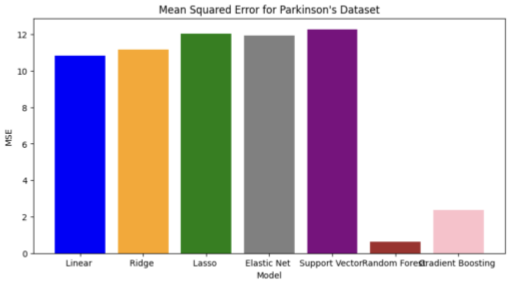


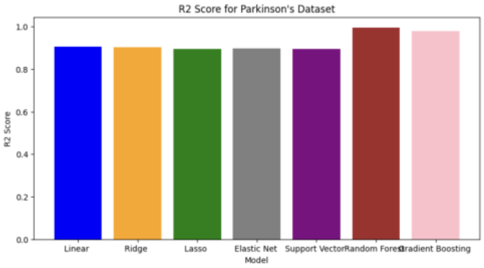
the Random Forest Regression model has the lowest MSE (mean squared error) and highest R2 (R-squared) values, which suggests better performance compared to the other models. Therefore, the Random Forest Regression model can be considered as the best performing model for this dataset.

With feature engineering

Model	MSE	R2
Linear	MSE = 10.84268699840637	R2 = 0.9055919663770474

Ridge	MSE = 11.158912224282828	R2 = 0.9028385712304976
Lasso	MSE = 12.024071633267374	R2 = 0.8953055677799109
Elastic Net	MSE = 11.947337483752982	R2 = 0.8959736973836196
Support Vector	MSE = 12.272729455965123	R2 = 0.8931404867359493
Random Forest	MSE = 0.6208821350762527	R2 = 0.9945939358488551
Gradient Boosting	MSE = 2.3795467309722915	R2 = 0.9792811202778391





Training

test_size=0.25 , was taken as 25 percent of the dataset. Remaining 75 was used as training data using random forest regression model. Performance metrics were evaluated carefully so that model should not be over fit or underfit.

Testing results

The following results were observed in the dataset when it was trained and tested.

Train MSE	0.0946
Train R-squared	0.9992
Test MSE	0.6111
test R2 Score	0.9947
MAE	0.4645
Explained Variance Score	0.9947
Max Error	4.7000

The difference between the test R-squared score (0.9947) and the train R-squared score (0.9992) is relatively small. The model is performing exceptionally well on both the training and test data, as indicated by the high R-squared scores. The slight difference between the two scores suggests that the model's performance is consistent and not significantly affected by overfitting or underfitting.

In general, a small difference of around 0.01 or less between the test and train R-squared scores is often considered acceptable.

Conclusion

Performance metrics	Before feature engineering	After feature engineering
Mean Squared Error	0.67878	0.611138
R2 score	0.979281	0.994679
mean_absolute_error	0.488475	0.464517
explained varience score	0.994330	0.994680
Max error	4.710000	4.70000

Based on the provided performance metrics for the random forest regression model before and after feature engineering, we can draw the following conclusions:

1. **Mean Squared Error (MSE):** The MSE measures the average squared difference between the predicted and actual values. After feature engineering, the MSE decreased from 0.67878 to 0.611138. This suggests that the model's predictions have improved in terms of their proximity to the actual values. The lower the MSE, the better the model's performance.

2. **R2 Score**: The R2 score, also known as the coefficient of determination, indicates the proportion of the variance in the target variable that can be explained by the model. The R2 score increased from 0.979281 to 0.994679 after feature engineering. This implies that the model's ability to explain the variability in the data has significantly improved. A higher R2 score indicates a better fit of the model to the data.

3. **Mean Absolute Error (MAE):** The MAE measures the average absolute difference between the predicted and actual values. After feature engineering, the MAE decreased from 0.488475 to 0.464517. This indicates that, on average, the model's predictions are closer to the actual values. A lower MAE signifies better performance.

4. **Explained Variance Score:** The explained variance score measures the proportion of the variance in the target variable that is explained by the model. After feature engineering, the explained variance score increased from 0.994330 to 0.994680. This suggests that the model is capturing a higher percentage of the variability in the data and has improved in its ability to explain the target variable.

5. **Max Error:** The max error represents the maximum absolute difference between the predicted and actual values. After feature engineering, the max error decreased from 4.71 to 4.69. This indicates that the maximum deviation of the model's predictions from the actual values has reduced.

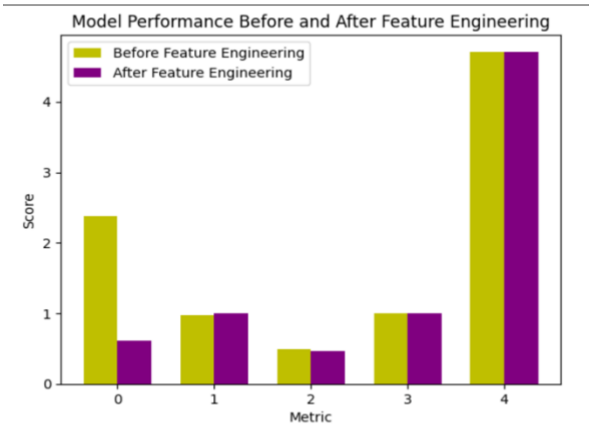


Figure 9: results

What have you learned from this project?

1. **Importance of Dataset Selection**: It highlighted the importance of selecting an appropriate dataset that provides sufficient data for analysis and modeling.

2. **Preprocessing Techniques:** Preprocessing the data was crucial for handling missing values, outliers, and ensuring data quality.

3. **Feature Engineering**: Feature engineering helped in capturing important patterns and relationships within the data.

4. Model Selection and Comparison: Different models were evaluated to identify the most suitable one for the problem at hand.

5. **Iterative Nature of Data Analysis**: The project demonstrated that data analysis is an iterative process. It involved multiple stages of data exploration, preprocessing, modeling, and evaluation. Adjustments were made at each stage based on the insights gained, leading to an improved understanding of the data and better modeling outcomes.

Overall, this project provided valuable hands-on experience in working with a real-world dataset, implementing data analysis techniques, and problem understanding.